**VIM**

Vim (a contraction of Vi IMproved) is a free and open-source, screen-based text editor program. It is an improved clone of Bill Joy's vi. Vim's author, Bram Moolenaar, derived Vim from a port of the Stevie editor for Amiga[6] and released a version to the public in 1991. Vim is designed for use both from a command-line interface and as a standalone application in a graphical user interface.

## Vim Modes:

1. Command Mode
2. Command-Line Mode
3. Insert Mode
4. Visual Mode

:edit new.txt

:write new.txt  
:set number

:noh

:s/cat/dog/

:s/cat/dog/g

:%s/dog/cat/g

:%s/dog/cat/gi

:0, 1 s/cat/lion/gi

/cat #find cat

A

Append text to end of current line.

I

Insert text at beginning of line.

o (lowercase letter “o”)

Open blank line below cursor for text.

O (uppercase letter “o”)

Open blank line above cursor for text.

s

Delete character at cursor and substitute text.

S

Delete line and substitute text.

R

Overstrike existing characters with new characters.

yy - yank (copy) a line

* 2yy - yank (copy) 2 lines

p - put (paste) the clipboard after cursor

* P - put (paste) before cursor

**u**

Undo

**Ctrl + r**

redo

x Delete one character

dd multiple times, it will delete multiple lines.

To delete a specific line

:10d

To delete line numbers 6 to 8

:6,8d

To delete all lines below the current line

:.,$d

To delete all the lines of the file

:%d

**Save vim customizations:**  
  
:echo $VIM

~/.vimrc

syntax on

set ignorecase

<https://vim.wikia.com/wiki/Example_vimrc>

"

" Description: A minimal, but feature rich, example .vimrc. If you are a

" newbie, basing your first .vimrc on this file is a good choice.

" If you're a more advanced user, building your own .vimrc based

" on this file is still a good idea.

"------------------------------------------------------------

" Features {{{1

"

" These options and commands enable some very useful features in Vim, that

" no user should have to live without.

" Set 'nocompatible' to ward off unexpected things that your distro might

" have made, as well as sanely reset options when re-sourcing .vimrc

set nocompatible

" Attempt to determine the type of a file based on its name and possibly its

" contents. Use this to allow intelligent auto-indenting for each filetype,

" and for plugins that are filetype specific.

if has('filetype')

filetype indent plugin on

endif

" Enable syntax highlighting

if has('syntax')

syntax on

endif

"------------------------------------------------------------

" Must have options {{{1

"

" These are highly recommended options.

" Vim with default settings does not allow easy switching between multiple files

" in the same editor window. Users can use multiple split windows or multiple

" tab pages to edit multiple files, but it is still best to enable an option to

" allow easier switching between files.

"

" One such option is the 'hidden' option, which allows you to re-use the same

" window and switch from an unsaved buffer without saving it first. Also allows

" you to keep an undo history for multiple files when re-using the same window

" in this way. Note that using persistent undo also lets you undo in multiple

" files even in the same window, but is less efficient and is actually designed

" for keeping undo history after closing Vim entirely. Vim will complain if you

" try to quit without saving, and swap files will keep you safe if your computer

" crashes.

set hidden

" Note that not everyone likes working this way (with the hidden option).

" Alternatives include using tabs or split windows instead of re-using the same

" window as mentioned above, and/or either of the following options:

" set confirm

" set autowriteall

" Better command-line completion

set wildmenu

" Show partial commands in the last line of the screen

set showcmd

" Highlight searches (use <C-L> to temporarily turn off highlighting; see the

" mapping of <C-L> below)

set hlsearch

" Modelines have historically been a source of security vulnerabilities. As

" such, it may be a good idea to disable them and use the securemodelines

" script, <http://www.vim.org/scripts/script.php?script\_id=1876>.

" set nomodeline

"------------------------------------------------------------

" Usability options {{{1

"

" These are options that users frequently set in their .vimrc. Some of them

" change Vim's behaviour in ways which deviate from the true Vi way, but

" which are considered to add usability. Which, if any, of these options to

" use is very much a personal preference, but they are harmless.

" Use case insensitive search, except when using capital letters

set ignorecase

set smartcase

" Allow backspacing over autoindent, line breaks and start of insert action

set backspace=indent,eol,start

" When opening a new line and no filetype-specific indenting is enabled, keep

" the same indent as the line you're currently on. Useful for READMEs, etc.

set autoindent

" Stop certain movements from always going to the first character of a line.

" While this behaviour deviates from that of Vi, it does what most users

" coming from other editors would expect.

set nostartofline

" Display the cursor position on the last line of the screen or in the status

" line of a window

set ruler

" Always display the status line, even if only one window is displayed

set laststatus=2

" Instead of failing a command because of unsaved changes, instead raise a

" dialogue asking if you wish to save changed files.

set confirm

" Use visual bell instead of beeping when doing something wrong

set visualbell

" And reset the terminal code for the visual bell. If visualbell is set, and

" this line is also included, vim will neither flash nor beep. If visualbell

" is unset, this does nothing.

set t\_vb=

" Enable use of the mouse for all modes

if has('mouse')

set mouse=a

endif

" Set the command window height to 2 lines, to avoid many cases of having to

" "press <Enter> to continue"

set cmdheight=2

" Display line numbers on the left

set number

" Quickly time out on keycodes, but never time out on mappings

set notimeout ttimeout ttimeoutlen=200

" Use <F11> to toggle between 'paste' and 'nopaste'

set pastetoggle=<F11>

"------------------------------------------------------------

" Indentation options {{{1

"

" Indentation settings according to personal preference.

" Indentation settings for using 4 spaces instead of tabs.

" Do not change 'tabstop' from its default value of 8 with this setup.

set shiftwidth=4

set softtabstop=4

set expandtab

" Indentation settings for using hard tabs for indent. Display tabs as

" four characters wide.

"set shiftwidth=4

"set tabstop=4

"------------------------------------------------------------

" Mappings {{{1

"

" Useful mappings

" Map Y to act like D and C, i.e. to yank until EOL, rather than act as yy,

" which is the default

map Y y$

" Map <C-L> (redraw screen) to also turn off search highlighting until the

" next search

nnoremap <C-L> :nohl<CR><C-L>

"------------------------------------------------------------

To go to a specific line number, **enter the line number in Command-line Mode (:<line number>) or the line number and G (<line number>G) in Normal Mode**.

A black and white text on a black background

Description automatically generated

**The count**

All movement command can be prefixed by typing a number, e.g., 5w to move the cursor  
five words, o r 12j to move t h e cursor 12 lines down. In fact. every single comm and ( including switching to insert mode) can be repeated a fixed number of times by typing the number of repeats before t h e actual command . In vim terminology, this is referred to as the **count**.

3 insert foo

**Replacing text**

change command

cw

cb

cc (Replace line)

ciw (Will replace the entire current word, not just from the current cursor position)

caw (Will do as ciw but include before and after spaces)

C (replace to the end of line)

r (just replace the character under the cursor)

~ (Replace the case of the character)

2d

2x

**Copy and Paste**

copy = yank

paste = put

4yaw will copy the current word and next three (for total of four)

p will put after the current cursor.  
P will put before the current cursor.

**Multiple register**

26 named registers

3”tyy

2”zyy

“tp

“zp

When the upper case version of a register is used, the text that is being cut or yanked i s appended to t hat register instead of overwriting it.

There are 10 numbered registers, " 0 t hrough " 9. Register " 0 wi l l always have a copy of t h e most  
recent yanked text, while register "1 will have a copy of the most recent deleted text. When new  
text is changed or deleted, the contents of "1 will shift in to " 2, " 2 into " 3, etc.

Pressing . ( period ) from command mode will redo the last edit action , but  
on the current line. This can be used to easily perform the same edit action multiple times.

BASH Copy and Paste:  
  
Ctrl u  
Ctrl y